Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2004 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2004

Web Services Security: Proposed Model for Content Delivery Assurance in a Low Trust Scenario

Nicole Lang Beebe
University of Texas at San Antonio

Roy Calvo
University of Texas at San Antonio

Srinivasan Rao
University of Texas at San Antonio

Web Services Security: Proposed Model for Content Delivery Assurance in a Low Trust Scenario

Nicole Lang Beebe[1]
University of Texas at San Antonio
nbeebe@utsa.edu

Roy Calvo
University of Texas at San Antonio
rcalvo@utsa.edu

V. Srinivasan Rao
University of Texas at San Antonio
crao@utsa.edu

ABSTRACT

Web Services Security (WS-Security) provides a set of standards as a basis for the development of security models to address the handling of SOAP messages. Scenarios explicating these standards have limited their focus to confidentiality, authenticity and integrity. The issue of delivery assurance in a low trust scenario has not been raised or addressed. In this article, we demonstrate the current standards are adequate to develop a security model that incorporates delivery assurance for a transaction model. Based on the lessons learned in this exercise, we argue that a theoretical approach is needed to establish the adequacy of the standards and develop appropriate security models. We suggest trust among participants and flexibility of information flow as two dimensions for inclusion in the theoretical analysis. More work is needed to determine if additional dimensions are needed to completely characterize the transaction models (scenarios).

Keywords

Web Services, WS-Security, security, delivery, assurance, confirmation, theoretical approach.

INTRODUCTION

Web Services environments and transaction models are expected to facilitate exponential growth in e-commerce and business to business (B2B) productivity, due to their ability to decrease business costs and increase consumer choice. A supporting cast of enabling technology has already been developed, most notably eXtensible Markup Language (XML) (Bray et al., 2004) and Simple Object Access Protocol (SOAP) (Gudgin et al., 2003). However, it is generally accepted that the inclusion of Web Services into current business models will not occur until security-related concerns are resolved.

The increased security risks associated with Web Services are mostly due to changes in the source of threat and asset exposure. In the conventional Internet environment, the primary sources of the threat are external, i.e., threats to the message during transit by parties unassociated with the transaction. In Web Services, there may be an additional source of threat, i.e., internal threat. Messages designated from one party to another may be routed through an intermediary. Hence, in scenarios in which there is there is low trust among participants, it is necessary to incorporate mechanisms to prevent tampering with, or deletion of, the message by the intermediary, i.e., safeguard against threats from parties who are a part of the overall transaction. The change in asset exposure arises because Web Services could allow untrusted communications into trusted networks.

Efforts to address new security concerns include the development of specifications, such as WS-Security, WS-SecurityPolicy, WS-Trust, WS-SecureConversation and so on. Additional standards (e.g. XML encryption, XML digital signature, Security Assertion Markup Language (SAML), XML Access Control Markup Language) address the finer granularity of requirements for security implementation. Collectively, these are intended to provide the basis for "a standards-based architecture that is comprehensive yet flexible enough to meet the Web services security needs of real businesses" (IBM and Microsoft, 2002).

[1] Author names are in alphabetical order. All authors have contributed equally.

"At the same time, every customer and every Web service has its own unique security requirements based upon their own particular business needs" (IBM and Microsoft, 2002). While some attention has been paid to the development of specific solutions for specific business needs, much work remains to be done in this area. One issue that has not received much attention is delivery assurance. Delivery assurance can be divided into two categories: assurance in spite of technology failures and assurance against the acts of untrustworthy human actors. Vendor proprietary reliable messaging protocols and the proposed reliable HTTP protocol (HTTP-R) address delivery assurance in spite of technology failures (Todd et al., 2002). These solutions are designed to ensure that the message is delivered only once, and the sender is notified in the event the receiver does not receive the message. It should be noted that these solutions are designed for two-party transactions, whereas most web-services scenarios involve multi-party transactions. Actions of untrustworthy human actors include: (a) repudiation by sender, (b) repudiation by receiver, (c) data alteration by an actor, and (d) deletion of messages designated for others by an intermediary. Repudiation by sender is addressed by the use of XML digital signature (Mactaggart, 2001). Repudiation by receiver is handled by WS-Non-Repudiation (Gravengaard, 2003). XML encryption along with hash algorithms enables the detection of data alteration or partial content deletion, but not entire message or element-level deletions within SOAP envelopes, i.e., the deletion of the content from A to D by intermediary B or C.

In this article, we have two goals. First, we provide a security model that incorporates delivery assurance in a particular scenario, as a first step in understanding issues related to delivery assurance and the development of more generalized solutions. Second, we propose a theoretical approach to the study of security issues. Towards this end, we offer two theoretical dimensions as a starting point.

CONCEPTUAL OVERVIEW

Web Services

The term, Web Services, encapsulates functional ideals that characterize a new way of conducting business via the Internet. While definitions vary, the common thread is that Web Services represent a conceptual idea more than a clearly defined entity or finite set of technologies. Gartner defines Web Services as, "loosely coupled software components that interact with one another dynamically via standard Internet technologies" (Myerson, 2002). Forrester research defines them as, "automated connections between people, systems, and applications that expose elements of business functionality as a software service and create new business value" (Myerson, 2002). In short, web services refer to a set of applications that work in heterogeneous, highly dynamic environments, wherein cross-platform interoperability is achieved via common communication standards (i.e. Simple Object Access Protocol (SOAP)), flexible messaging languages (i.e. XML), and ubiquitous protocols (i.e. Hyper Text Transfer Protocol (HTTP)). Web services models display the following characteristics:

- Relationships between the partners are dynamic, transient, and not established via *a priori* agreements;
- Middleware and common messaging protocols are leveraged to remove client-server code dependencies;
- Transactions often occur via asynchronous messaging, rather than traditional session-based connections;
- Transactions may include a third party intermediary (or multiple intermediaries);
- Transactions leverage peer-to-peer architectures (Westbridge, 2002, 2003); and
- Architectures require decentralized administration (Westbridge, 2002, 2003)

Web Services Security Requirements

Incremental security requirements in Web services, beyond the traditional needs of confidentiality, integrity, authentication, access control and non-repudiation, arise from the transient nature of the relationships between the participants. First, in traditional Internet-based business environments, the threats to security are from external sources, i.e., parties unrelated to the transactions. In Web services environments, since messages are often routed through intermediaries, threats from internal sources may also exist. These raise the need for end-to-end security and element level confidentiality. Second, threats from internal sources go beyond the customary confidentiality, integrity, and authenticity issues. There is a possibility that messages intended for other recipients may be fully or partially deleted by intermediaries. Third, communications related to Web services are allowed through firewalls of organizations as a matter of course. This provides a point of entry for malicious agents, like viruses into organizational networks.

Web Services Security Specifications and Technology

In recognition of the need for new security model enabling standards and technology, IBM™, Microsoft™, and VeriSign™ drafted the Web Services Security (WS-Security) specification in 2002 (Kaler, 2002). Using WS-Security and supporting

specifications mentioned in this article, businesses are able to develop basic security models to fit various scenarios and environments. Such models emphasize message integrity, element-level confidentiality, single-message authentication, and extensible security token exchange in Web Services transactions. The stated goals of the specification are to ensure that an antagonist cannot read or modify SOAP messages without authorization, nor can they forge SOAP messages that will be processed as if authentic. Its drafters facilitate interoperability in a security context by outlining a set of SOAP extensions that provide security token information, handle digital signatures (including signatures for canonicalized data), and encryption/decryption mechanisms for body blocks, header blocks, or any combination thereof.

Unfulfilled Security Requirements

WS-Security is limited in scope, and its authors acknowledge that it does not establish a security context or authentication mechanisms that require multiple exchanges, nor does it facilitate key exchange and derived keys, nor does it address how trust is established or determined. The first two are addressed via the WS-SecureConversation specification (Kaler and Nadalin, 2002), while trust establishment is facilitated in part by WS-SecurityPolicy (Hondo and Kaler, 2003) and WS-Trust (Kaler and Nadalin, 2002) specifications. WS-Security is limited in scope in other ways important to e-commerce, including assurance that messages are delivered to intended recipients when intermediaries are involved in the transaction.

Delivery Assurance of Targeted Content

In a typical Web service scenario, several participants (A, B, C and so on) are all involved in a transaction, but have different roles and therefore differing needs regarding sending and receiving various parts of the overall communication. Assuming participant A is the initiating party to the transaction, A may have some information for B and different information for C—some or all of which may be private (not privy to B). Additionally, B may have its own unique information to send to C in addition to forwarding information from A (whether or not B has access to the message from A to C). WS-Security and supporting specifications and technologies protect the confidentiality and integrity of A's message to C, but fail to prevent B from deleting A's message to C entirely. Participant C would have no way of knowing whether requisite information from the transaction was supposed to have come from A instead of B, nor any way of knowing whether or not C should have received supplemental information from A through B. This concern becomes significant when there is limited trust among participants.

Consider the hypothetical case of an individual wanting a vacation package that includes airline flights, hotel accommodation, and excursions arranged by a travel agent. Currently, the customer receives limited options for each of these services from the travel agent, and makes a choice. Travel agents often bundle services from companies that provide the best financial incentives to the travel agent. In a web-services scenario, the customer could efficiently insist that the travel agent contact specific airlines, hotels and tour companies, in order to compare prices with the standard travel agent package. Each actor in the sequence has a reason to delete messages intended for the next receiver. The travel agent may have special arrangements with a particular airline, and may therefore not wish to the customer to see lower quotes. Likewise, special arrangements may exist with hotels and tour companies. The customer does not wish to contact the airline, hotel or tour company directly because each actor has a role, or specific information that has to be passed on to the next actor. The travel agent's role is to put the package together; the airline has to confirm availability dates and pass the arrival/departure information to the hotel, which in turn has to pass on location information to the tour company to ensure that the tour company serves that area. Further, the customer may not wish the airline to know what type of room is being requested, or the hotel to know the types of tours that the customer is interested in.

This scenario is not an exemplar of a realistic scenario, based on current business processes. Instead it reflects what could be implemented in a web services environment to provide options to the customer, options, which are not currently feasible without a lot of effort on the part of the customer. Whether such a scenario is realistic or not, it reflects an extreme in terms of requirements. A solution, which can handle the extreme requirements, will be able to handle more relaxed requirements.

Proposed Solution

Literature Review

Little has addressed delivery assurance in Web Services transactions characterized by low levels of trust among participants. WS-Security, WS-SecurePolicy, WS-Trust, and WS-SecureConversation do not address non-repudiation in the form of message delivery assurance. The World Wide Web Consortium (W3C) addresses the issue of communicating with third party or multiple intermediaries in its discussion of Web Services architectures (Haas and Orchard, 2002). The third party intermediary scenario is built around a buyer-seller scenario in which the third party intermediary is a marketplace. In this scenario, buyers and sellers establish trusted relationships with the marketplace, which then brokers deals between them. Current security mechanisms fulfill the security needs in this type of scenario, not to mention the fact that the marketplace is

motivated to act in a forthright manner since their relationships with buyers and sellers is key to their continued business viability. Such motivation cannot be assumed in a low trust environment and not all relationships involving third party intermediaries will be established a priori.

In the scenario of multiple intermediaries, the W3C proposes a solution wherein non-repudiation is achieved by intermediaries logging and signing routing header information in a persistent, presumably third party, database. Such logging and signing is handled by a "Routing Logging Handler" on each SOAP node. Such a model, however, only provides non-repudiation at the message level. It does not provide protection against, or, detection of, intermediaries removing content in its possession intended for a subsequent recipient.

Introduction to Proposed Security Model

The proposed security model is developed for a particular, if somewhat artificial scenario. We begin by describing the scenario and defining the terminology used in the security model. We outline the current approach to such a scenario and indicate the lack of delivery assurance. Then we explicate our security model for the same scenario, and how it addresses delivery assurance.

Scenario

Our discussion of the solution is based on the following scenario. The number of participants in the transaction is known, four in this case, A, B, C and D. Participant A initiates the transaction. The information from A is passed along a known sequence, from A to B to C to D. Participant A has distinct messages for each of the three other participants. The messages from A to each of the participants must be kept confidential from the others. Also, each participant must be assured of the integrity of the message and that it is authentically from A. Participant A needs to be sure that each of the participants receives A's message (delivery assurance).

At this point in time, we will assume that B does not have an independent message to transmit to C or D.

Terminology

We use the following terminology in our discussion:

- The term "message" refers to the primary content that one party wishes to convey to another in its plaintext format. M_{AB} represents the message from A to B.
- The hashed version of the message is represented by HM_{AB}.
- The public key of A in the asymmetric encryption scheme is represented by PUK_A.
- The private key of A in the asymmetric encryption scheme is represented by PRK_A.
- The symmetric key shared by A and B is represented by SK_{AB}.
- A payload refers to an aggregation of components being sent from one party to another, which only the receiving party can access. Payload from A to B will be represented by P_{AB}. A typical payload from A to B would include, M_{AB} and HM_{AB} encrypted with the PRK_A for authentication, and then encrypted with PUK_B for confidentiality. In addition to M_{AB} and $HM_{AB,}$ other content may be included in the payload, as will be seen later in this article. Specific contents of payload will have to be specified in the security model.
- The SOAP envelope from A to B will include P_{AB} and will include payloads to other parties in the transaction in a multi-party transaction, i.e., P_{AC} and P_{AD} will also be present in the SOAP envelope from A to B. However, P_{AC} and P_{AD} will be encrypted so that B is not able to violate the confidentiality of communication between A and C, and, A and D.
- The proof-of-completion token, T_{DA}, is the token that the end-point D returns to A to indicate that the transaction is complete.

Basic Security Model Currently Employed

Current security models rely on the trust among the participants. The basic solution uses three different payloads, one for each of the recipients. The payload from A to B is as follows: $[PUK_B\{PRK_A(M_{AB} + HM_{AB})\}]$. In words, the message from A to B (M_{AB}) and its hash digest (HM_{AB}) are encrypted using A's private key (PRK_A), and then again using B's public key (PUK_B). Confidentiality is assured, since the message must first be decrypted with B's private key (PRK_B). Authenticity is

established when A's public key successfully decrypts at the next stage. Integrity can be verified with the hash digest. Similar payloads are created for C and D.

Participant A creates a SOAP envelope, which contains the three payloads, one for B, one for C, and one for D, and sends the envelope to B. Participant B opens the envelope, extracts the payload from A to B, and reconstitutes the SOAP envelope with the payloads from A to C and A to D, and forwards it to C. The process continues until the payload for the final recipient is delivered.

In the process of opening the SOAP envelope from A and reconstituting it for transmission to C, B can delete either the payload to C or D or both. It is possible that the characteristics of the transaction are such that the absence of participation from one of the participants may be noticed. It is also possible that the deletion may go totally undetected. In essence, detection of the failure of delivery is dependent on chance, or some other manual, human-in-the-loop effort to confirm all messages were delivered and received as intended.

Proposed Solution[2]: Nested Symmetric Keys Security Model

The proposed solution is based on three principles. First, a sequential dependence is created between adjacent intermediaries in the sequence. The nature of the dependency is such that each intermediary can view his/her payload only if he/she has access to the appropriate symmetric key used to encrypt his/her message. This symmetric key can be provided only by the actor immediately ahead of him/her in the sequence. Second, the final participant (end-point) must return a "proof-of-completion" token to the party initiating the transaction, a token that has to be enclosed in the payload for the final participant (also referred to as end-point) from the initiator. Third, the mandatory payload principle requires that A must have a payload for each of the subsequent participants. In the case where there is no message, a null message must be sent. Sequential dependence, proof-of-completion token and mandatory payloads will ensure that no participant can be deliberately or accidentally left out of the sequence without immediate detection.

Next we describe our solution. The SOAP envelope from A to B will include three payloads: A→B, A→C, and A→D. The three payloads are shown symbolically first and then described in words.

a) Payload from A to B: $PUK_B[PRK_A \{SK_{AB}(M_{AB}) + HM_{AB}\} + \{SK_{AB}\}] + PUK_B[\{PUK_C(SK_{AC} + \{PUK_D(SK_{AD} + T_{DA})\})\}]$
b) Payload from A to C: $PUK_C[PRK_A \{SK_{AC}(M_{AC}) + HM_{AC}\}]$
c) Payload from A to D: $PUK_D[PRK_A \{SK_{AD}(M_{AD}) + HM_{AD}\}]$

The payload from A to B has two components. In the first component, the message (M_{AB}) is encrypted using a symmetric key (SK_{AB}). The encrypted message along with the hash digest, HM_{AB} is signed using the private key of A (PRK_A). The first component also has the symmetric key (SK_{AB}) used to encrypt the message from A to B. The second component has the symmetric key for A to D messages (SK_{AD}) and the proof-of-completion, T_{DA}, encrypted with D's public key (PUK_D). This is combined with the symmetric key for A to C messages (SK_{AC}), and then encrypted with the public key for C. Each of the components is encrypted separately with the public key for B.

The payload from A to C has only one component. The message M_{AC} is encrypted with SK_{AC}, which along with the hash digest, HM_{AC}, is encrypted using the private key of A (PRK_A), and then encrypted with the public key of C (PUK_C). The payload from A to D is similar to the payload from A to C.

Participant B, upon receiving the SOAP envelope will use its private key to decrypt the symmetric key and the authenticated message plus digest package. Upon accessing the encrypted message, B will use the symmetric key to get at the message. Participant B will also use its private key on the second component.

The reconstituted SOAP envelope from B to C will include:

a) The encrypted symmetric keys: $PUK_C(SK_{AC} + \{PUK_D(SK_{AD} + T_{DA})\})$
b) Payload from A to C: $PUK_C[PRK_A \{SK_{AC}(M_{AC}) + HM_{AC}\}]$
c) Payload from A to D: $PUK_D[PRK_A \{SK_{AD}(M_{AD}) + HM_{AD}\}]$

Participant C, upon receiving the SOAP envelope from B, will apply its private key to the encrypted symmetric keys and extract the symmetric key, and decrypt the payload, verify its authenticity, decrypt the message with the symmetric key and then verify its integrity.

[2] The XML code for this scenario is available from Prof. Rao.

The reconstituted SOAP envelope from C to D will include:

a) The encrypted symmetric keys: $PUK_D(SK_{AD}+T_{DA})$
b) Payload from A to D: $PUK_D[PRK_A \{SK_{AD}(M_{AD}) + HM_{AD}\}]$

Participant D, upon receiving the SOAP envelope will decrypt the key, and access D's message. The proof-of-completion token is also accessible, and can be returned to A, signed with D's private key.

Examination of the scheme will reveal that B cannot skip C, because C has to decrypt SK_{AD}, before D can access it. If B deletes the message from A to C, and sends the SOAP envelope to C for the decryption of D's symmetric key, the absence of a payload from A to C will alert C. If B deletes the message to D, once again C will be aware that a payload has been deleted. If C deletes the payload from A to D and the symmetric key package, the proof-of-completion token will be lost, and hence A will be alerted.

Further examination will show that it is possible for B, or any other intermediary, to initiate a similar sequence of messages, and send it in parallel to A's messages in the same SOAP envelopes, and be assured that the overall process is reliable and foolproof. Solutions can be similarly formulated for the 3-actor or n-actor scenarios. When only three actors are involved, several feasible solutions exist, including one analogous to the example shown for the 4-actor scenario.

MODEL LIMITATIONS

Some caveats are in order. First, we are focused on conceptually demonstrating technical feasibility of a solution. Our solution does not address organizational or social issues, which may still undermine the overall security of the system. Second, the solution is based on existing encryption technologies, in particular, the asymmetric key concept. The solution is only as sound as the technologies that are used in its development and the veracity of the infrastructure that facilitates it (i.e. a Public Key Infrastructure (PKI)). Third, the model assumes that the initiator knows the number of participants and the sequence in which they must receive the messages. We acknowledge that in the publish-find-bind scenario of web services, this assumption is easily challengeable. We use it as a starting point for model development and agree that more complex scenarios will have to be addressed to provide significant practical value. Fourth, the issue of low trust has only been partially addressed. Our solution is focused on providing safeguards against message deletion by internal sources. Our model does not address the threats arising from allowing untrustworthy participants access through the firewall. Fifth, the proposed solution requires repeated encryption and decryption, and reconstitution of SOAP envelopes, which places demands on processing resources. The goal in this article was to demonstrate that a feasible solution existed. Future research will have to examine ways improve the efficiency of processing.

FURTHER RESEARCH

We selected one single transaction model (scenario) and developed a solution that ensures delivery assurance (or detection of its violation) without using a central repository. Security models for other transaction models can be developed and similarly examined, on an as-needed basis. From a practical perspective, this may be sufficient to further the development of Web services. From a theoretical perspective, the challenge is to demonstrate that the standards are sufficient to develop a solution for any transaction model.

We propose a theoretical approach for consideration. First, transaction models can be classified along some dimensions. In the segment following, we propose two dimensions. We are not sufficiently advanced in our research to make a categorical statement that the dimensions are adequate to completely specify transaction models. Second, we argue that security demands are more difficult to fulfill at one end of each dimension than the other. Hence, if security models can be developed for the extreme conditions, then it would be possible to claim, based on theoretical demonstration, that the existing standards are adequate to meet the security needs of Web services.

Theoretical Dimensions

In our example, we made two explicit assumptions: first, there was low trust among the participants, and second, that the number of participants and the sequence in which they receive the payloads were known, i.e., the information flow was inflexible. These constitute two of the dimensions that can guide the theoretical approach.

Trust among participants

In low trust situations, steps have to be taken against violations of confidentiality, authenticity, integrity and deletion of payloads by participants in the transaction. Chance detection of data deletion is possible, but a secure system should

incorporate active mechanisms to detect deletion of payloads or abortion of the process sequence. Low or zero trust assumption makes stronger demands on security requirements. We argue that if a security solution can be formulated for all cases of zero trust, then theoretically, the security mechanisms incorporated into the current standards are sufficient to handle all transaction models.

Flexibility of Information Flow

Flexibility of information flow is a function of three factors: number of participants, linearity of transmission of messages, and predetermination of the sequence of transmission. The most complex case (highest flexibility of information flow) would be when the number of participants is not known at the beginning of the message transmission, i.e., intermediaries may introduce new participants into the group; the transmission of messages is non-linear, i.e., the same or different messages may be transmitted along different paths; the sequence of transmission is not known, i.e., the identity of the $(n)^{th}$ person receiving it is determined by the $(n-1)^{th}$ party, which determination is not known to the first (n-2) parties. Security models that can deal with high flexibility of information flow will be able to deal with situations of low flexibility in information flow.

In the scenario that we considered, trust among participants was low (high security demands), but flexibility of information flow was low (low security demands). Thus, we have addressed a scenario with intermediate security demands. In terms of addressing the theoretical adequacy of the standards, one still needs to demonstrate that security models exist which can handle the need for high flexibility in information flow utilizing the current standards.

We have articulated two theoretical dimensions. Further analysis is needed to examine if other relevant dimensions exist.

CONCLUSIONS

The concept of Web services introduces of the need for delivery assurance. In this article, a security model for delivery assurance for a specific scenario was presented. Two dimensions, trust among participants, and, flexibility of information flow, were proposed as a starting point for a theoretical approach to examining the adequacy of existing security standards and mechanisms.

ACKNOWLEDGEMENTS

A summer research grant from the Center for Infrastructure Assurance and Security, UTSA, facilitated Prof. Rao's participation in this research.

REFERENCES

1. Albrecht, Conan C. (2004) How Clean is the Future of SOAP?, *Communications of the ACM*, 47, 2, pp 66-68
2. Ben-Itzhak, Yuval (2003) Securing WebServices, *The ISSA Journal*, October 2003
3. Bray, Tim, Jean Paoli, C. M. Sperberg-McQueen, Eve Maler, Francois Yergeau, and John Cowan (2004) Extensible Markup Language (XML) 1.1—W3C Recommendation 4 February 2004, http://www.w3.org/TR/xml11/
4. Gravengaard, Eric L. (editor) (2003) Web Services Security: Non-Repudiation—Proposal Draft 05, 11 April 2003, http://xml.coverpages.org/ReactivityWS-NonRepudiation-05.pdf
5. Gudgin, Martin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik Frystyk Nielsen (2003) SOAP Version 1.2 Part 1: Messaging Framework—W3C Recommendation 24 June 2003, http://www.w3.org/TR/soap12-part1/
6. Haas, Hugo and David Orchard (editors) (2002) Web Services Architecture Usage Scenarios: W3C Working Draft 30 July 2002, http://www.w3.org/TR/2002/WD-ws-arch-scenarios-20020730
7. Hondo, Maryann and Chris Kaler (2003) Web Services Policy Framework (WS-Policy), Version 1.1 28 May 2003, http://msdn.microsoft.com/library/en-us/dnglobspec/html/ws-policy.asp?frame=true
8. Hughes, John and Eve Maler (editors) (2004) Technical Overview of the OASIS Security Assertion Markup Language (SAML) V1.1, Draft 04, 30 March 2004, http://www.oasis-open.org/committees/download.php/6193/sstc-saml-tech-overview-1.1-draft-04.pdf
9. IBM and Microsoft (2002). Security in Web Services World: A Proposed Architecture and Roadmap, Version 1.0, April 7, 2002, http://www-106.ibm.com/developerworks/webservices/library/ws-secmap/
10. Kaler, Chris (editor) (2002) Specification: Web Services Security (WS-Security), Version 1.0 05 April 2001, http://www-106.ibm.com/developerworks/webservices/library/ws-secure
11. Kaler, Chris and Anthony Nadalin (editors) (2002) Specification: Web Services Trust Language (WS-Trust), Draft 18 December 2002, http://www.ibm.com/developerworks/library/ws-trust/index.html

12. Kaler, Chris and Anthony Nadalin (editors) (2002) Specification: Web Services Secure Conversation (WS-SecureConversation), Draft 18 December 2002, http://www-106.ibm.com/developerworks/library/ws-secon
13. Mackey, Dick (2003) Web Services Security, *The ISSA Journal,* September 2003
14. Mactaggart, Murdoch (2001) Enabling XML Security—An Introduction to XML Encryption and XML Signature, http://www-106.ibm.com/developerworks/xml/library/s-xmlsec.html/index.html
15. Madsen, Paul (2003) WS-Trust: Interoperable Security for Web Services, http://www.xml.com/puba/ws/2003/06/24/ws-trust.html
16. Miyazawa, Tatsuo and Takayuki Kushida (2000) An Advanced Internet XML/EDI Model Based on Secure XML Documents.
17. Myerson, Judith M. (2002) Web Services Architectures: How They Stack Up, *Web Services Architect*, 23 January 2002, http://www.webservicesarchitect.com/content/articles/myerson01.asp
18. Naedele, Martin (2003) Standards for XML and Web Services Security, *Computer*, April 2003
19. Todd, Stephen, Francis Parr, and Michael H. Conner (2002) A Primer for HTTPR—An Overview of the reliable HTTP protocol, 1 July 2001, Updated 1 April 2002, http://www-106.ibm.com/developerworks/webservices/library/ws-phtt/
20. Westbridge (2002) XML Web Services Security: Going Production, http://www.westbridgetech.com
21. Westbridge (2003) Guide to XML: Web Services Security, http://www.westbridgetech.com